UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
10/688,640	10/17/2003	Antti Kokkinen	14322US02	1958

23446 7590 12/30/2008

MCANDREWS HELD & MALLOY, LTD
500 WEST MADISON STREET
SUITE 3400
CHICAGO, IL 60661

EXAMINER	
WANG, BEN C	

ART UNIT	PAPER NUMBER
2192	

MAIL DATE	DELIVERY MODE
12/30/2008	PAPER

Please find below and/or attached an Office communication concerning this application or proceeding.

The time period for reply, if any, is set in the attached communication.

Office Action Summary	Application No.	Applicant(s)	
	10/688,640	KOKKINEN, ANTTI	
	Examiner	Art Unit	
	BEN C. WANG	2192	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE *3* MONTH(S) OR THIRTY (30) DAYS, WHICHEVER IS LONGER, FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on *21 October 2008*.

2a)☐ This action is **FINAL**. 2b)☒ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) *1-24* is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) *1-24* is/are rejected.

7)☐ Claim(s) _____ is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☐ The specification is objected to by the Examiner.

10)☐ The drawing(s) filed on _____ is/are: a)☐ accepted or b)☐ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____.

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1) ☒ Notice of References Cited (PTO-892)
2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)
3) ☐ Information Disclosure Statement(s) (PTO/SB/08) Paper No(s)/Mail Date _____.
4) ☐ Interview Summary (PTO-413) Paper No(s)/Mail Date. _____.
5) ☐ Notice of Informal Patent Application
6) ☐ Other: _____.

DETAILED ACTION

1. Applicant's amendment dated October 21, 2008, responding to the Non-Final Office action mailed July 18, 2008 provided in the rejection of claims 1-24.

Claims 1-24 remain pending in the application and which have been fully considered by the examiner.

Applicant's arguments with respect to claims rejection have been fully considered but are moot in view of the new grounds of rejection – see *Chiang* and *Multer et al.* - arts made of record, as applied hereto.

Claim Rejections – 35 USC § 102(e)

The following is quotation of 35 U.S.C. 102(e) which form the basis for all obviousness rejections set forth in this office action:

> (e) the invention was described in (1) an application for patent, published under section 122(b), by another filed in the United States before the invention by the applicant for patent or (2) a patent granted on an application for patent by another filed in the United States before the invention by the applicant for patent, except that an international application filed under the treaty defined in section 351(a) shall have the effects for purposes of this subsection of an application filed in the United States only if the international application designated the United States and was published under Article 21(2) of such treaty in the English language.

2. Claims 1-7 and 9-11 are rejected under 35 U.S.C. 102(e) as being anticipated by Ying-Hsin Robert Chiang (Pub. No. US 2004/0062130 A1) (hereinafter 'Chiang' – art made of record)

3. **As to claim 1** (Previously Presented), Chiang discloses a method for updating software in an electronic device (Abstract - … An upgrade client of a remote device

receives a delta file block that codes differences between an original and a new version of a file ...), the method comprising:

- generating an update package for updating at least one software application being generated based upon difference information between the at least one software application and at least one reference software installed on the electronic device (e.g., Fig. 1 (server side - 102), elements 110 – ORIGINAL FILE (i.e., a reference software); 112 – NEW FILE (i.e., a software application); 116 – DELTA FILE (i.e., difference information – an update package); 114 – BYTE-LEVEL FILE DIFFERENCING ALGORITHM; [0029] – The file differencing algorithm 114 receives the new file 112, compares it to the original file 110, and calculates the byte-level differences between the compared files ... The file differencing algorithm 114 generates a difference file 116, referred to herein as a delta file, during the comparison);
- updating the at least one software application using the update package and the reference software (e.g., Fig. 1 - 110 (client side - 104) – HOSTED ORIGINAL FILE (i.e., the reference software); 116 – DELTA FILE (i.e., the update package); 112 – COPY OF NEW FILE (updated version); BYTE-LEVEL FILE UPDATING ALGORITHM; [0031] - ... The file updating algorithm 118 hosted on the receiving computer system 104 uses the delta file 116 along of the new file 112. This copy of the new file 112 is then used to update the original file 110 hosted on the client device 104 that is targeted for revision or updating ...); and

- wherein the updating leaves the at least one reference software unchanged (e.g., Abstract, Lines 5-8 – The upgrade client writes an original file block corresponding to the delta file block from an original memory area to a second memory area; Fig. 6, step 602 – Copy original file from original ROM location into reserved ROM space; Fig. 7)

4. **As to claim 2** (Original) (incorporating the rejection in claim 1), Chiang discloses the method wherein generating an update package for updating the at least one software application based upon the at least one reference software installed on the electronic device comprises:

- accessing a copy of the at least one reference software;
- retrieving a copy of the at least one software application; and
- generating an update package (e.g., Fig. 1 (server side - 102), elements 110 – ORIGINAL FILE (i.e., a reference software); 112 – NEW FILE (i.e., a software application); 116 – DELTA FILE (i.e., difference information – an update package); 114 – BYTE-LEVEL FILE DIFFERENCING ALGORITHM; [0029] – The file differencing algorithm 114 receives the new file 112, compares it to the original file 110, and calculates the byte-level differences between the compared files … The file differencing algorithm 114 generates a difference file 116, referred to herein as a delta file, during the comparison)

5. **As to claim 3** (Original) (incorporating the rejection in claim 1), Chiang discloses the method wherein generating an update package for updating at least one software application based upon the at least one reference software installed on the electronic device comprises:

- accessing a copy of the at least one reference software;
- retrieving a copy of each of multiple versions of the at least one software application; and
- generating an update package comprising all transitions between the retrieved versions of the at least one software application (e.g., Fig. 1 (server side - 102), elements 110 – ORIGINAL FILE (i.e., a reference software); 112 – NEW FILE (i.e., a software application); 116 – DELTA FILE (i.e., difference information – an update package); 114 – BYTE-LEVEL FILE DIFFERENCING ALGORITHM; [0029] – The file differencing algorithm 114 receives the new file 112, compares it to the original file 110, and calculates the byte-level differences between the compared files … The file differencing algorithm 114 generates a difference file 116, referred to herein as a delta file, during the comparison; [0030] - … The delta file 116 includes meta-data along with actual data of replacement and/or insertion operations that represent the differences between the new or current version of the associated file and previous versions of the file …)

6. **As to claim 4** (Original) (incorporating the rejection in claim 1), Chiang discloses the method further comprising updating multiple update versions of the at least one

software application installed on the electronic device is performed using a single update package (e.g., Fig. 1 (server side - 102), elements 110 – ORIGINAL FILE (i.e., a reference software); 112 – NEW FILE (i.e., any updated version software); 116 – DELTA FILE (i.e., difference information – an update package); 114 – BYTE-LEVEL FILE DIFFERENCING ALGORITHM; [0029] – The file differencing algorithm 114 receives the new file 112, compares it to the original file 110, and calculates the byte-level differences between the compared files ... The file differencing algorithm 114 generates a difference file 116, referred to herein as a delta file, during the comparison)

7. **As to claim 5** (Original) (incorporating the rejection in claim 1), Chiang discloses the method further comprising installing the at least one software application and the at least one reference software as part of a single installation (e.g., Fig. 1 - 110 (client side - 104) – HOSTED ORIGINAL FILE (i.e., the reference software); 116 – DELTA FILE (i.e., the update package); 112 – COPY OF NEW FILE (updated version); BYTE-LEVEL FILE UPDATING ALGORITHM; [0031] - ... The file updating algorithm 118 hosted on the receiving computer system 104 uses the delta file 116 along of the new file 112. This copy of the new file 112 is then used to update the original file 110 hosted on the client device 104 that is targeted for revision or updating ...)

8. **As to claim 6** (Original) (incorporating the rejection in claim 1), Chiang discloses the method further comprising updating the at least one reference software and updating the at least one software application (e.g.,) as part of a single update (e.g., Fig.

1 - 110 (client side - 104) – HOSTED ORIGINAL FILE (i.e., the reference software); 116 – DELTA FILE (i.e., the update package); 112 – COPY OF NEW FILE (updated version); BYTE-LEVEL FILE UPDATING ALGORITHM; [0031] - ... The file updating algorithm 118 hosted on the receiving computer system 104 uses the delta file 116 along of the new file 112. This copy of the new file 112 is then used to update the original file 110 hosted on the client device 104 that is targeted for revision or updating ...)

9. **As to claim 7** (Original) (incorporating the rejection in claim 1), Chiang discloses the method wherein the at least one software application comprises a plurality of software applications, and the at least one reference software comprises a plurality of reference software (e.g., Fig. 1 (server side - 102), elements 110 – ORIGINAL FILE (i.e., a reference software); 112 – NEW FILE (i.e., a software application); 116 – DELTA FILE (i.e., difference information – an update package); 114 – BYTE-LEVEL FILE DIFFERENCING ALGORITHM; [0029] – The file differencing algorithm 114 receives the new file 112, compares it to the original file 110, and calculates the byte-level differences between the compared files ... The file differencing algorithm 114 generates a difference file 116, referred to herein as a delta file, during the comparison; [0030] - ... The delta file 116 includes meta-data along with actual data of replacement and/or insertion operations that represent the differences between the new or current version of the associated file and previous versions of the file ...)

10. **As to claim 9** (Original) (incorporating the rejection in claim 7), Chiang discloses the method further comprising:

- identifying a software application needing updating from the plurality of software applications installed on the electronic device (e.g., [0036] - … the upgrade system 200 maintains embedded software components on client devices 104 via a wireless connection with the device 212, thereby enabling wireless carriers to continuously provide the latest data services to all users); and
- identifying whether a reference software corresponding to the software application needing updating is present on the electronic device, wherein if the reference software is present then retrieving an update package for the software application needing updating (e.g., [0031] – The file updating algorithm 118 hosted on the receiving computer system 104 uses the delta file 116 along with the hosted original file 110 to generate or create a copy of the new file 112 …);
- verifying the update package (e.g., [0043] – In response to receipt of the confirmation from the handset, the upgrade server 204 authenticates and authorizes the user and/or requesting device, and verifies prerequisite capabilities and limitations of the requesting device …); and
- installing the update package on the electronic device (e.g., [0033] - … The upgrade system uses the delta file and file updating algorithm of an embodiment in supporting software maintenance and application management for client device ...)

11. **As to claim 10** (Original) (incorporating the rejection in claim 7), Chiang discloses the method further comprising:

- identifying a software application needing updating from the plurality of software applications installed on the electronic device (e.g., [0036] - ... the upgrade system 200 maintains embedded software components on client devices 104 via a wireless connection with the device 212, thereby enabling wireless carriers to continuously provide the latest data services to all users);
- determining if the update is needed immediately; and
- storing the update until the update is needed immediately (e.g., [0058] – The client device uses another procedure when the new file is a critical software component ... a flow diagram 500 of critical software component ...)

12. **As to claim 11** (Original) (incorporating the rejection in claim 10), Chiang discloses the method wherein when the update is determined to be needed immediately, then

- invoking an update agent to employ at least the stored update package and reference software (e.g., Fig. 1 - 110 (client side - 104) – HOSTED ORIGINAL FILE (i.e., the reference software); 116 – DELTA FILE (i.e., the update package); 112 – COPY OF NEW FILE (updated version); BYTE-LEVEL FILE UPDATING ALGORITHM; [0031] - ... The file updating algorithm 118 hosted on the receiving computer system 104 uses the delta file 116 along of the new file 112. This copy

of the new file 112 is then used to update the original file 110 hosted on the client device 104 that is targeted for revision or updating ...); and

- updating the software application with the update package (e.g., [0033] - ... The upgrade system uses the delta file and file updating algorithm of an embodiment in supporting software maintenance and application management for client device ...)

Claim Rejections – 35 USC § 103(a)

The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

13. Claims 8 and 12-24 are rejected under 35 U.S.C. 103(a) as being unpatentable over Chiang in view of Multer et al. (Pat. No. US 6,694,336 B1) (hereinafter 'Multer' – art made of record)

14. **As to claim 8** (Original) (incorporating the rejection in claim 7), Chiang discloses the method further comprising:

- identifying a software application needing updating from the plurality of software applications installed on the electronic device (e.g., [0036] - ... the upgrade system 200 maintains embedded software components on client devices 104 via

a wireless connection with the device 212, thereby enabling wireless carriers to continuously provide the latest data services to all users)

Further, Chiang discloses a system and method for updating electronic files and file components are provided (e.g., Abstract), but does not explicitly disclose other limitations stated below.

However, in an analogous art of *Data Transfer and Synchronization System*, Multer discloses the following:

- identifying whether a reference software corresponding to the software application needing updating is present on the electronic device, wherein if the reference software is not present, then installing the software application and an associated reference software in a single update on the electronic device (e.g., Fig. 8; Col. 9, Lines 1-20 - ... Distribution of the device engines may occur via, for example, an installation package forwarded over an Internet connection ... The processing load associated with delivering this service is pushed to the end-point devices which provides for easy scaling of the system to ever-larger application; Col. 15, Line 57 - Col. 16, Line 5)

Therefore, it would have been obvious to one of ordinary skill in the art, at the time the invention was made to combine the teachings of Multer into the Chiang's system to further provide other limitations stated above in the Chiang system.

The motivation is that it would further enhance the Chiang's system by taking, advancing and/or incorporating the Multer's system which offers significant advantages that applications which efficiently synchronize data between disparate types of devices

can provide advantages in applications beyond synchronizing individual, person information between a personal information manager hardware device and a personal computer as once suggested by Multer (e.g., Col. 2, Line 63 – Col. 3, Line 7)

15. **As to claim 12** (Previously Presented), Chiang discloses a system for updating software (Abstract - ... An upgrade client of a remote device receives a delta file block that codes differences between an original and a new version of a file ...), the system comprising:

- an electronic device capable of having software installed thereon (e.g., Fig. 2, element 206 – Upgrade Client); and
- the software delivery device receiving and delivering at least one update package to the electronic device, wherein the at least one update package is based on differences between at least one application software and the reference software, and the reference software facilitates (e.g., Fig. 1 (server side - 102), elements 110 – ORIGINAL FILE (i.e., a reference software); 112 – NEW FILE (i.e., a software application); 116 – DELTA FILE (i.e., difference information – an update package); 114 – BYTE-LEVEL FILE DIFFERENCING ALGORITHM; [0029] – The file differencing algorithm 114 receives the new file 112, compares it to the original file 110, and calculates the byte-level differences between the compared files ... The file differencing algorithm 114 generates a difference file 116, referred to herein as a delta file, during the comparison), using the at least one update package, at least one update to the application software installed on the

electronic device (e.g., Fig. 1 - 110 (client side - 104) – HOSTED ORIGINAL FILE (i.e., the reference software); 116 – DELTA FILE (i.e., the update package); 112 – COPY OF NEW FILE (updated version); BYTE-LEVEL FILE UPDATING ALGORITHM; [0031] - ... The file updating algorithm 118 hosted on the receiving computer system 104 uses the delta file 116 along of the new file 112. This copy of the new file 112 is then used to update the original file 110 hosted on the client device 104 that is targeted for revision or updating ...), and wherein the updating leaves the reference software unchanged (e.g., Abstract, Lines 5-8 – The upgrade client writes an original file block corresponding to the delta file block from an original memory area to a second memory area; Fig. 6, step 602 – Copy original file from original ROM location into reserved ROM space; Fig. 7)

Further, Chiang discloses a system and method for updating electronic files and file components are provided (e.g., Abstract), but does not explicitly disclose other limitations stated below.

However, in an analogous art of *Data Transfer and Synchronization System*, Multer discloses the following:

- a software delivery device for receiving and installing a reference software to the electronic device if the electronic device does not have the reference software previously installed (e.g., Fig. 8; Col. 9, Lines 1-20 - ... Distribution of the device engines may occur via, for example, an installation package forwarded over an Internet connection ... The processing load associated with delivering this service

is pushed to the end-point devices which provides for easy scaling of the system to ever-larger application; Col. 15, Line 57 - Col. 16, Line 5)

Therefore, it would have been obvious to one of ordinary skill in the art, at the time the invention was made to combine the teachings of Multer into the Chiang's system to further provide other limitations stated above in the Chiang system.

The motivation is that it would further enhance the Chiang's system by taking, advancing and/or incorporating the Multer's system which offers significant advantages that applications which efficiently synchronize data between disparate types of devices can provide advantages in applications beyond synchronizing individual, person information between a personal information manager hardware device and a personal computer as once suggested by Multer (e.g., Col. 2, Line 63 – Col. 3, Line 7)

16. **As to claim 13** (Original) (incorporating the rejection in claim 12), Chiang discloses the system wherein the electronic device further comprises an update agent, the update agent being capable of employing the reference software in conjunction with any retrieved update package to generate updated versions of the application software and also being capable of updating a plurality of application software employing reference software associated with each application software (e.g., Fig. 1 - 110 (client side - 104) – HOSTED ORIGINAL FILE (i.e., the reference software); 116 – DELTA FILE (i.e., the update package); 112 – COPY OF NEW FILE (updated version); BYTE-LEVEL FILE UPDATING ALGORITHM; [0031] - … The file updating algorithm 118 hosted on the receiving computer system 104 uses the delta file 116 along of the new

file 112. This copy of the new file 112 is then used to update the original file 110 hosted on the client device 104 that is targeted for revision or updating ...; [0030] - ... The delta file 116 includes meta-data along with actual data of replacement and/or insertion operations that represent the differences between the new or current version of the associated file and previous versions of the file ...)

17. **As to claim 14** (Original) (incorporating the rejection in claim 12), Chiang discloses the system further comprising an update generating system, the update generating system comprising a loader manager, the loader manager:

- managing loading of application software and application software version updates from the software delivery device (e.g., [0033] - ... The upgrade system uses the delta file and file updating algorithm of an embodiment in supporting software maintenance and application management for client devices ...);
- employing a loader from a loader module (e.g., [0054] - ... this process can be repeated until all dependency files are also loaded into the memory; P. 11, Left-Col., Lines 31-37); and
- employing security services to authenticate software being delivered (e.g., [0043] – In response to receipt of the confirmation from the handset, the upgrade server 204 authenticates and authorizes the user and/or requesting device, and verifies prerequisite capabilities and limitations of the requesting device ...)

18. **As to claim 15** (Original) (incorporating the rejection in claim 14), Chiang discloses the system wherein the loader manager further comprises an installation

agent for installing application software and downloading files from the software delivery device (e.g., [0062] - … The upgrade client then downloads the delta file corresponding to one block of the target EBSC to be updated from the upgrade server into the client device RAM … Following writing of the new EBSC block to RAM ...)

19. **As to claim 16** (Original) (incorporating the rejection in claim 16), Chiang discloses the system wherein the loader manager is adapted to:

- identify an application software needing updating (e.g., [0036] - … the upgrade system 200 maintains embedded software components on client devices 104 via a wireless connection with the device 212, thereby enabling wireless carriers to continuously provide the latest data services to all users);
- identify whether reference software associated with the application software needing updating exists (e.g., [0045] – The client device determines the status of numerous device parameters prior to participating in an update procedure …); and
- coordinating an update of the application software and an associated reference software in a single update (e.g., Fig. 1 - 110 (client side - 104) – HOSTED ORIGINAL FILE (i.e., the reference software); 116 – DELTA FILE (i.e., the update package); 112 – COPY OF NEW FILE (updated version); BYTE-LEVEL FILE UPDATING ALGORITHM; [0031] - … The file updating algorithm 118 hosted on the receiving computer system 104 uses the delta file 116 along of the

new file 112. This copy of the new file 112 is then used to update the original file 110 hosted on the client device 104 that is targeted for revision or updating ...)

20. **As to claim 17** (Original) (incorporating the rejection in claim 14), Chiang discloses the system wherein the loader manager is adapted to:

- retrieve the update package (e.g., [0033] - ... The upgrade system uses the delta file and file updating algorithm of an embodiment in supporting software maintenance and application management for client devices ...);
- access contents of the update package (e.g., [0062] - ... The upgrade client then downloads the delta file corresponding to one block of the target EBSC to be updated from the upgrade server into the client device RAM ...); and
- verify the update package (e.g., [0043] – In response to receipt of the confirmation from the handset, the upgrade server 204 authenticates and authorizes the user and/or requesting device, and verifies prerequisite capabilities and limitations of the requesting device ...)

21. **As to claim 18** (Original) (incorporating the rejection in claim 14), Chiang discloses the system wherein the loader manager is adapted to determine immediacy of a needed update for a particular application software (e.g., [0058] – The client device uses another procedure when the new file is a critical software component ... a flow diagram 500 of critical software component ...)

22. **As to claim 19** (Original) (incorporating the rejection in claim 12), Chiang discloses the system wherein the software delivery device is one of a server, a CDROM, and a network (e.g., Fig. 2, elements 203 – Software Component Certification Server; 212)

23. **As to claim 20** (Original) (incorporating the rejection in claim 12), Chiang discloses the system wherein the electronic device is one of a computer, a digital phone, and a digital camera (e.g., Fig. 2, element 104)

24. **As to claim 21** (Original), Chiang discloses a method for updating software in an electronic device (Abstract - ... An upgrade client of a remote device receives a delta file block that codes differences between an original and a new version of a file ...) the method comprising:

- generating a first update package for updating at least one software application, the first update package being generated based upon difference information between first and second software versions (e.g., Fig. 1 (server side - 102), elements 110 – ORIGINAL FILE (i.e., a first software version); 112 – NEW FILE (i.e., a second software version); 116 – DELTA FILE (i.e., difference information – an update package); 114 – BYTE-LEVEL FILE DIFFERENCING ALGORITHM; [0029] – The file differencing algorithm 114 receives the new file 112, compares it to the original file 110, and calculates the byte-level differences between the

compared files ... The file differencing algorithm 114 generates a difference file 116, referred to herein as a delta file, during the comparison); and

- generating a second update package for updating the at least one software application, the second update package being generated based upon difference information between first and third software versions (e.g., Fig. 1 (server side - 102), elements 110 – ORIGINAL FILE (i.e., a first software version); 112 – NEW FILE (i.e., a third software version); 116 – DELTA FILE (i.e., difference information – an update package); 114 – BYTE-LEVEL FILE DIFFERENCING ALGORITHM; [0029] – The file differencing algorithm 114 receives the new file 112, compares it to the original file 110, and calculates the byte-level differences between the compared files ... The file differencing algorithm 114 generates a difference file 116, referred to herein as a delta file, during the comparison);

Further, Chiang discloses a system and method for updating electronic files and file components are provided (e.g., Abstract) but does not explicitly disclose other limitations stated below.

However, in an analogous art of *Data Transfer and Synchronization System*, Multer discloses the following:

- generating a third update package for updating the at least one software application, the third update package being generated based upon difference information between the first and second update packages (e.g., Col. 39, Lines 62-67 - ... data packages can be merged into larger meta-data packages. Such meta-data information, such as the

organization of multiple device packages, may be encoded into a larger system package. Each system package is essentially an encoded sequence of data packages)

Therefore, it would have been obvious to one of ordinary skill in the art, at the time the invention was made to combine the teachings of Multer into the Chiang's system to further provide other limitations stated above in the Chiang system.

The motivation is that it would further enhance the Chiang's system by taking, advancing and/or incorporating the Multer's system which offers significant advantages that applications which efficiently synchronize data between disparate types of devices can provide advantages in applications beyond synchronizing individual, person information between a personal information manager hardware device and a personal computer as once suggested by Multer (e.g., Col. 2, Line 63 – Col. 3, Line 7)

Furthermore, Chiang discloses updating the at least one software application using the third update package (e.g., Fig. 1 - 110 (client side - 104) – HOSTED ORIGINAL FILE (i.e., the reference software); 116 – DELTA FILE (i.e., the update package); 112 – COPY OF NEW FILE (updated version); BYTE-LEVEL FILE UPDATING ALGORITHM; [0031] - ... The file updating algorithm 118 hosted on the receiving computer system 104 uses the delta file 116 along of the new file 112. This copy of the new file 112 is then used to update the original file 110 hosted on the client device 104 that is targeted for revision or updating ...)

25. **As to claim 22** (Original), Chiang discloses a method for updating software in an electronic device (Abstract - … An upgrade client of a remote device receives a delta file block that codes differences between an original and a new version of a file …), the method comprising:

- generating a first update package for updating at least one software application, the first update package being generated based upon difference information between a first software version and a reference software corresponding to the at least one software application (e.g., Fig. 1 (server side - 102), elements 110 – ORIGINAL FILE (i.e., a reference software); 112 – NEW FILE (i.e., a first software version); 116 – DELTA FILE (i.e., difference information – an update package); 114 – BYTE-LEVEL FILE DIFFERENCING ALGORITHM; [0029] – The file differencing algorithm 114 receives the new file 112, compares it to the original file 110, and calculates the byte-level differences between the compared files … The file differencing algorithm 114 generates a difference file 116, referred to herein as a delta file, during the comparison);
- generating a second update package for updating the at least one software application, the second update package being generated based upon difference information a second software version and the reference software corresponding to the at least one software application (e.g., Fig. 1 (server side - 102), elements 110 – ORIGINAL FILE (i.e., a reference software); 112 – NEW FILE (i.e., a second software version); 116 – DELTA FILE (i.e., difference information – an update package); 114 – BYTE-LEVEL FILE DIFFERENCING ALGORITHM;

[0029] – The file differencing algorithm 114 receives the new file 112, compares it to the original file 110, and calculates the byte-level differences between the compared files ... The file differencing algorithm 114 generates a difference file 116, referred to herein as a delta file, during the comparison);

Further, Chiang discloses a system and method for updating electronic files and file components are provided (e.g., Abstract) but does not explicitly disclose other limitations stated below.

However, in an analogous art of *Data Transfer and Synchronization System*, Multer discloses:

- generating a third update package for updating the at least one software application, the third update package being generated based upon difference information between the first and second update packages (e.g., Col. 39, Lines 62-67 - ... data packages can be merged into larger meta-data packages. Such meta-data information, such as the organization of multiple device packages, may be encoded into a larger system package. Each system package is essentially an encoded sequence of data packages)

Therefore, it would have been obvious to one of ordinary skill in the art, at the time the invention was made to combine the teachings of Multer into the Chiang's system to further provide other limitations stated above in the Chiang system.

The motivation is that it would further enhance the Chiang's system by taking, advancing and/or incorporating the Multer's system which offers significant advantages that applications which efficiently synchronize data between disparate types of devices

can provide advantages in applications beyond synchronizing individual, person information between a personal information manager hardware device and a personal computer as once suggested by Multer (e.g., Col. 2, Line 63 – Col. 3, Line 7)

Furthermore, Chiang discloses updating the at least one software application using the third update package (e.g., Fig. 1 - 110 (client side - 104) – HOSTED ORIGINAL FILE (i.e., the reference software); 116 – DELTA FILE (i.e., the update package); 112 – COPY OF NEW FILE (updated version); BYTE-LEVEL FILE UPDATING ALGORITHM; [0031] - … The file updating algorithm 118 hosted on the receiving computer system 104 uses the delta file 116 along of the new file 112. This copy of the new file 112 is then used to update the original file 110 hosted on the client device 104 that is targeted for revision or updating …)

26. **As to claim 23** (Original), Chiang discloses a system for updating software (Abstract - … An upgrade client of a remote device receives a delta file block that codes differences between an original and a new version of a file …), the system comprising:

- an electronic device capable of having software installed thereon (e.g., Fig. 2, element 206 – Upgrade Client);
- a first update package generator for generating update packages based upon difference information between different versions of software (e.g., Fig. 1 (server side - 102), elements 110 – ORIGINAL FILE (i.e., version 1 software); 112 – NEW FILE (i.e., version 2 software); 116 – DELTA FILE (i.e., difference information – an update package); 114 – BYTE-LEVEL FILE DIFFERENCING

ALGORITHM; [0029] – The file differencing algorithm 114 receives the new file 112, compares it to the original file 110, and calculates the byte-level differences between the compared files … The file differencing algorithm 114 generates a difference file 116, referred to herein as a delta file, during the comparison);

Further, Chiang discloses a system and method for updating electronic files and file components are provided (e.g., Abstract) but does not explicitly disclose other limitations stated below.

However, in an analogous art of *Data Transfer and Synchronization System*, Multer discloses the following:

- a second update package generator for generating update packages based upon difference information between different update packages (e.g., Col. 39, Lines 62-67 - … data packages can be merged into larger meta-data packages. Such meta-data information, such as the organization of multiple device packages, may be encoded into a larger system package. Each system package is essentially an encoded sequence of data packages)

Therefore, it would have been obvious to one of ordinary skill in the art, at the time the invention was made to combine the teachings of Multer into the Chiang's system to further provide other limitations stated above in the Chiang system.

The motivation is that it would further enhance the Chiang's system by taking, advancing and/or incorporating the Multer's system which offers significant advantages that applications which efficiently synchronize data between disparate types of devices can provide advantages in applications beyond synchronizing individual, person

information between a personal information manager hardware device and a personal computer as once suggested by Multer.

Furthermore, Chiang discloses a software delivery device for delivering at least one update package generated based upon difference information between different update packages to the electronic device (e.g., Fig. 2; [0033] - … The upgrade system uses the delta file and file updating algorithm of an embodiment in supporting software maintenance and application management for client devices including mobile electronic devices …)

27. **As to claim 24** (Original), Chiang discloses a system for updating software (Abstract - … An upgrade client of a remote device receives a delta file block that codes differences between an original and a new version of a file …), the system comprising:

- an electronic device capable of having software installed thereon (e.g., Fig. 2, element 206 – Upgrade Client);
- a first update package generator for generating update packages based upon difference information between a version of software and a reference software corresponding to at least one software application (e.g., Fig. 1 (server side - 102), elements 110 – ORIGINAL FILE (i.e., a reference software); 112 – NEW FILE (i.e., a version software); 116 – DELTA FILE (i.e., difference information – an update package); 114 – BYTE-LEVEL FILE DIFFERENCING ALGORITHM; [0029] – The file differencing algorithm 114 receives the new file 112, compares it to the original file 110, and calculates the byte-level differences between the

compared files ... The file differencing algorithm 114 generates a difference file 116, referred to herein as a delta file, during the comparison);

Further, Chiang discloses a system and method for updating electronic files and file components are provided (e.g., Abstract) but does not explicitly disclose other limitations stated below.

However, in an analogous art of *Data Transfer and Synchronization System*, Multer discloses the following:

- a second update package generator for generating update packages based upon difference information between different update packages (e.g., Fig. 8; Col. 9, Lines 1-20 - ... Distribution of the device engines may occur via, for example, an installation package forwarded over an Internet connection ... The processing load associated with delivering this service is pushed to the end-point devices which provides for easy scaling of the system to ever-larger application; Col. 15, Line 57 - Col. 16, Line 5)

Therefore, it would have been obvious to one of ordinary skill in the art, at the time the invention was made to combine the teachings of Multer into the Chiang's system to further provide other limitations stated above in the Chiang system.

The motivation is that it would further enhance the Chiang's system by taking, advancing and/or incorporating the Multer's system which offers significant advantages that applications which efficiently synchronize data between disparate types of devices can provide advantages in applications beyond synchronizing individual, person

information between a personal information manager hardware device and a personal computer as once suggested by Multer (e.g., Col. 2, Line 63 – Col. 3, Line 7)

Furthermore, Chiang discloses a software delivery device for delivering at least one update package generated based upon difference information between different update packages to the electronic device (e.g., Fig. 1 - 110 (client side - 104) – HOSTED ORIGINAL FILE (i.e., the reference software); 116 – DELTA FILE (i.e., the update package); 112 – COPY OF NEW FILE (updated version); BYTE-LEVEL FILE UPDATING ALGORITHM; [0031] - … The file updating algorithm 118 hosted on the receiving computer system 104 uses the delta file 116 along of the new file 112. This copy of the new file 112 is then used to update the original file 110 hosted on the client device 104 that is targeted for revision or updating …)

Conclusion

28. Any inquiry concerning this communication or earlier communications from the examiner should be directed to Ben C. Wang whose telephone number is 571-270-1240. The examiner can normally be reached on Monday - Friday, 8:00 a.m. - 5:00 p.m., EST.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Tuan Q. Dam can be reached on 571-272-3695. The fax phone number for the organization where this application or proceeding is assigned is 571-273-8300.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free). If you would like assistance from a USPTO Customer Service Representative or access to the automated information system, call 800-786-9199 (IN USA OR CANADA) or 571-272-1000.

/Ben C Wang/
Ben C. Wang
Examiner, Art Unit 2192

/Tuan Q. Dam/
Supervisory Patent Examiner, Art Unit 2192